Practitioner's Docket No. NAI1P069/99.074.01 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Handong Wu et al.

Application No.: 09/609,690 Group No.: 2157
Filed: 07/05/2000 Examiner: Gold, Avi M.
For: HIGH PERFORMANCE PACKET PROCESSING USING A GENERAL PURPOSE PROCESSOR

Mail Stop Appeal Briefs – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF
(PATENT APPLICATION--37 C.F.R. § 41.37)

1. This brief is in furtherance of the Notice of Appeal, filed in this case on 11/20/2007, which reinstates the appeal originally instated by the Notice of Appeal filed on 09/27/2004, and the original Appeal Brief filed 09/29/2004.

2. STATUS OF APPLICANT

   This application is on behalf of other than a small entity.

3. FEE FOR FILING APPEAL BRIEF

   Pursuant to 37 C.F.R. § 41.20(b)(2), the fee for filing the Appeal Brief is:

other than a small entity	$510.00
Appeal Brief fee due	**$510.00**

4. EXTENSION OF TERM

The proceedings herein are for a patent application and the provisions of 37 C.F.R. § 1.136 apply.

Applicant believes that no extension of term is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

5. TOTAL FEE DUE

The total fee due is:

Appeal brief fee	$0.00 (previously paid on 09/29/2004)
Extension fee (if any)	$0.00
TOTAL FEE DUE	**$0.00**

6. FEE PAYMENT

Applicant believes that no fees are due in connection with the filing of this paper because the appeal brief fee was paid with a previous submission. However, the Commissioner is authorized to charge any additional fees that may be due (e.g. for any reason including, but not limited to, fee changes, etc.) to Deposit Account No. 50-1351 (Order No. NAI1P069).

7. FEE DEFICIENCY

If any additional extension and/or fee is required, and if any additional fee for claims is required, charge Deposit Account No. 50-1351 (Order No. NAI1P069).

Date: January 22, 2008

/KEVINZILKA/
Signature of Practitioner

Reg. No.: 41,429
Tel. No.: 408-971-2573
Customer No.: 28875

Kevin J. Zilka
Zilka-Kotab, PC
P.O. Box 721120
San Jose, CA 95172-1120

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	
Wu et al.	Group Art Unit: 2157
Application No. 09/609,690	Examiner: Gold, Avi M.
Filed: 07/05/2000	Atty. Docket No. NAI1P069/99.074.01
For: HIGH PERFORMANCE PACKET PROCESSING USING A GENERAL PURPOSE PROCESSOR	Date: 01/22/2008

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

ATTENTION: Board of Patent Appeals and Interferences

APPEAL BRIEF (37 C.F.R. § 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on 11/20/2007, which reinstates the appeal originally instated by the Notice of Appeal filed on 09/27/2004, and the original Appeal Brief filed 09/29/2004.

The fees required under § 1.17, and any required petition for extension of time for filing this brief and fees therefor, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

This brief contains these items under the following headings, and in the order set forth below (37 C.F.R. § 41.37(c)(i)):

I REAL PARTY IN INTEREST
II RELATED APPEALS AND INTERFERENCES
III STATUS OF CLAIMS
IV STATUS OF AMENDMENTS

V SUMMARY OF CLAIMED SUBJECT MATTER

VI GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

VII ARGUMENT

VIII CLAIMS APPENDIX

IX EVIDENCE APPENDIX

X RELATED PROCEEDING APPENDIX

The final page of this brief bears the practitioner's signature.

I REAL PARTY IN INTEREST (37 C.F.R. § 41.37(c)(1)(i))

The real party in interest in this appeal is McAfee, Inc.

II RELATED APPEALS AND INTERFERENCES (37 C.F.R. § 41.37(c) (1)(ii))

With respect to other prior or pending appeals, interferences, or related judicial proceedings that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, prior appeals were noted on 09/26/2005 and 09/27/2004 in the present application.

A Related Proceedings Appendix is appended hereto.

III STATUS OF CLAIMS (37 C.F.R. § 41.37(c) (1)(iii))

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-18, and 30

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims withdrawn from consideration: None
2. Claims pending: 1-18, and 30
3. Claims allowed: None
4. Claims rejected: 1-18, and 30
5. Claims cancelled: 19-29

C. CLAIMS ON APPEAL

The claims on appeal are: 1-18, and 30

See additional status information in the Appendix of Claims.

IV STATUS OF AMENDMENTS (37 C.F.R. § 41.37(c)(1)(iv))

As to the status of any amendment filed subsequent to final rejection, the Amendment submitted on 07/29/2004 was entered by the Examiner.

V SUMMARY OF CLAIMED SUBJECT MATTER (37 C.F.R. § 41.37(c)(1)(v))

With respect to a summary of Claim 1, as shown in Figure 1 et al., an apparatus for processing data packets comprises a first data processing unit (e.g. see item 10 of Figure 1, etc.) adapted to filter incoming packets. Further, the apparatus comprises an addressable memory unit (e.g. see item 100 of Figure 1 etc.) in which a plurality of instruction sets for packet processing are stored. In addition, the apparatus comprises a second data processing unit (e.g. see item 20 of Figure 1, etc.) adapted to process incoming packets according to one of the plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit. Also, the apparatus comprises a data bus (e.g. see item 30 of Figure 1, etc.) connecting the addressable memory unit and the first and second data processing units. See, for example, page 2, lines 14-20; page 4, lines 11-12; and page 4, lines 15-16 et al.

With respect to a summary of Claim 30, as shown in Figure 1 et al., an apparatus for processing data packets comprises a first data processing unit (e.g. see item 10 of Figure 1, etc.) adapted to filter incoming packets. Further, the apparatus comprises an addressable memory unit (e.g. see item 100 of Figure 1, etc.) in which a plurality of instruction sets for packet processing are stored. In addition, the apparatus comprises a second data processing unit (e.g. see item 20 of Figure 1, etc.) adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit (e.g. see item 10 of Figure 1, etc.). Furthermore, the apparatus comprises a data bus (e.g. see item 30 of Figure 1, etc.) connecting the addressable memory unit and the first and second data processing units.

Additionally, a policy condition table (e.g. see item 50 of Figure 1, etc.) is connected to the first data processing unit, the policy condition table having a plurality of rules stored therein. Also, a policy action table (e.g. see item 76 of Figure 1, etc.) is connected to the data bus and the addressable memory unit, where the policy action table stores at least one data processing policy. Further, the first data processing unit comprises logic for matching a first incoming packet (e.g. see item 122 of Figure 1, etc.) to a stored first rule (e.g. see item 52 of Figure 1, etc.), and for generating a first thread (e.g. see item 232 of Figure 1, etc.) if the first incoming packet matches the first rule. The first thread identifies the location of one of the at least one data processing

policies in the policy action table (e.g. see item 76 of Figure 1, etc.). The second data processing unit (e.g. see item 20 of Figure 1, etc.) is adapted to process the first incoming packet according to the data processing policy corresponding to the first thread.

In addition, the data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in the addressable memory unit (e.g. see item 100 of Figure 1, etc.). The state block is used by said first set of instructions for processing the first incoming packet (e.g. see item 122 of Figure 1, etc.). Additionally, the first processing unit (e.g. see item 10 of Figure 1, etc.) further comprises logic for matching a second incoming packet (e.g. see item 124 of Figure 1, etc.) to a stored second rule (e.g. see item 54 of Figure 1, etc.), and for generating a second thread (e.g. see item 234 of Figure 1, etc.) if the second incoming packet matches the second rule. The second thread identifies the location of one of the at least one data processing policy in the policy action table (e.g. see item 76 of Figure 1, etc.). Further, the second data processing unit (e.g. see item 20 of Figure 1, etc.) is adapted to process the second incoming packet according to the data processing policy corresponding to the second thread.

Also, a memory unit is connected to the first data processing unit (e.g. see item 10 of Figure 1, etc.) and to the second data processing unit (e.g. see item 20 of Figure 1, etc.). The memory unit is adapted to temporarily store packets before processing by the second data processing unit. Also, the second data processing unit (e.g. see item 20 of Figure 1, etc.) comprises a plurality of general purpose processors for executing instructions in parallel. Additionally, the apparatus includes a control logic unit (e.g. see item 10 of Figure 1, etc.) coupled to an input and the policy condition table for feeding an arithmetic logic unit (e.g. see item 20 of Figure 1, etc.), which is in turn coupled to the policy action table and the state block for generating an output. See, for example, page 2, lines 14-20; page 3, lines 11-15; page 4, lines 11-16; page 5, lines 8-10; page 6, lines 4-19; page 7, line 18 – page 8, line 5; page 9, lines 13-14; and page 9, line 26 – page 10, line 19 et al.

Of course, the above citations are merely examples of the above claim language and should not be construed as limiting in any manner.

VI GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL (37 C.F.R. § 41.37(c)(1)(vi))

Following, under each issue listed, is a concise statement setting forth the corresponding ground of rejection.

Issue # 1: The Examiner has rejected Claim 1 under 35 U.S.C. 102(e) as being anticipated by Irwin (U.S. Patent No. 6,393,026).

Issue # 2: The Examiner has rejected Claims 2-16, and 30 under 35 U.S.C. 103(a) as being unpatentable over Irwin (U.S. Patent No. 6,393,026), in view of Kadambi et al. (U.S. Patent No. 6,850,521).

Issue # 3: The Examiner has rejected Claims 17, and 18 under 35 U.S.C. 103(a) as being unpatentable over Kadambi et al. (U.S. Patent No. 6,850,521), in view of Scales (U.S. Patent No. 5,761,729).

VII ARGUMENT (37 C.F.R. § 41.37(c)(1)(vii))

The claims of the groups noted below do not stand or fall together. In the present section, appellant explains why the claims of each group are believed to be separately patentable.

Issue # 1:

The Examiner has rejected Claim 1 under 35 U.S.C. 102(e) as being anticipated by Irwin (U.S. Patent No. 6,393,026).

Group #1: Claim 1

With respect to Claim 1, the Examiner has relied on the following excerpts from Irwin to meet appellant's claimed "a first data processing unit adapted to filter incoming packets." Specifically, the Examiner has argued that "Irwin discloses a data packet processing system that forwards packets and assigns a program counter."

> "The data packet processing is performed by executing a packet switching software of the router. The processing for forwarding a data packet can be broken into a number of procedures. Typically, packet forwarding requires procedures, such as translation of a header of the data packet to identify suitable routes, classification of a packet flow according to source and destination addresses..." (Col. 5, lines 46-52 - emphasis added)
>
> "[serv]ice route pruning, metric route pruning, policy route pruning, option processing, congestion control, scheduling, and performance monitoring." (Col. 5, lines 53-55 - not specifically cited - emphasis added)
>
> "As each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system." (Col. 6, lines 5-8)

Appellant respectfully asserts that the excerpts from Irwin relied upon by the Examiner merely disclose that "[t]he processing for forwarding a data packet can be broken into a number of procedures such as translation of a header of the data packet to identify suitable routes, classification of a packet flow according to source and destination addresses, type of service route pruning, metric route pruning, policy route pruning, option processing, congestion control,

scheduling, and performance monitoring" (emphasis added). However, merely disclosing that forwarding a data packet may be broken into procedures such as the translation of a header to identify suitable routes, and classification of a packet flow according to source and destination addresses, as in Irwin, fails to suggest "a first data processing unit adapted to filter incoming packets" (emphasis added), as claimed by appellant. Clearly, procedures for forwarding a data packet, as in Irwin, simply fails to even suggest "filter[ing] incoming packets," as claimed by appellant.

Additionally, appellant respectfully asserts that it would be unobvious to modify Irwin to "filter incoming packets" (emphasis added), as claimed by appellant. Specifically, Irwin teaches "a router having forwarding engines at network interfaces" where "[e]ach interface 12 may be provisioned with a forwarding engine 18 that processes incoming packets in order to determine which outgoing network interface 12 needs to be accessed through the interconnect structure of the switching fabric 14" (Col. 1, lines 32-40 – emphasis added). Further, Irwin teaches that "[s]ince the network processor 16 is no longer required to make routing decisions for a data packet, the packet throughput is capable of being scaled with the number of physical interfaces 12" such that "[t]he single point of congestion as formerly experienced with the centralization of the routing algorithms is replaced by multiple processors 12" (Col. 1, lines 52-57 – emphasis added).

Appellant asserts that Irwin's router with forwarding engines at each network interface simply utilizes the forwarding engines to determine which outgoing network interface needs to be accessed through the interconnect structure in order to alleviate the network processor from making routing decisions, which allows the throughput to be scaled with each interface. Clearly, a router with a forwarding engine at each interface that determines which outgoing network interface needs to be accessed through the interconnect structure, would be unable to filter incoming packets due to the congestion as formerly experienced with the centralization of the routing algorithms, as expressly noted in Irwin. Therefore, appellant asserts that it would be unobvious to include filtering, in the manner claimed by appellant, in Irwin's multiple network interface router.

Further, with respect to Claim 1, the Examiner has relied on the following excerpts from Irwin to meet appellant's claimed "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit." Specifically, the Examiner has argued that "Irwin discloses a thread assigned to the packets that allows the forwarding program to process the packets."

> "As each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system. The program counter uniquely defines a main program flow in the main forwarding program that is executed for the packet to select an output queue of the router for routing the packet. Stepping through the program counter temporarily creates a virtual CPU execution unit, or a hardware thread, for the data packet. Each thread is represented by a thread identification number which is defined by the program counter and a register file used by the process followed for packet processing." (Col. 6, lines 5-16 - emphasis added)

Appellant respectfully disagrees and asserts that the excerpt from Irwin relied upon by the Examiner merely teaches that "[a]s each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system" and that "[s]tepping through the program counter temporarily creates a virtual CPU execution unit, or a hardware thread, for the data packet" (emphasis added).

However, the mere disclosure of assigning a program counter to the packet, such that stepping through the program counter creates a virtual CPU execution unit or hardware thread for the data packet, as in Irwin, fails to teach "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit" (emphasis added), as claimed by appellant. Clearly, stepping through the program counter to create a hardware thread for the packet, as in Irwin, simply fails to even suggest "process[ing] incoming packets according to one of said plurality of instruction sets **after the filtering**, based on a thread" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Irwin, as argued above, simply fails to suggest any sort of "filtering," as claimed by appellant, let alone in the specific context claimed by appellant. For example, appellant asserts that Irwin merely discloses that "[w]hen the end procedural call is read and the end of the main forwarding program has been reached (S32), if there is no data packet to be processed (S34), the thread is placed in the queue of idle threads since the packet processing has been completed and the packet has been forwarded to the designated route" (Col. 10, lines 56-61 – emphasis added). Further, Irwin teaches that "[i]f there is a data packet to be processed (S34), the thread is reactivated for a new packet (S02)" (Col. 10, lines 61-62).

However, disclosing that the thread is placed in the queue of idle threads after the packet processing has been completed and the packet has been forwarded to the designated route, as in Irwin, fails to suggest "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit" (emphasis added), as claimed. Clearly, placing the thread in the queue of idle threads after processing is completed, as in Irwin, simply fails to even suggest "process[ing] incoming packets according to one of said plurality of instruction sets after **the filtering**, **based on a thread assigned** to the incoming packets by the first data processing unit" (emphasis added), as claimed by appellant.

In addition, with respect to Claim 1, the Examiner has relied on the following excerpts from Irwin to meet appellant's claimed "a data bus connecting the addressable memory unit and the first and second data processing units." Specifically, the Examiner has argued that "Irwin discloses a processor bus."

> "The received requests are forwarded to a processing unit (not shown) via a processor bus 119 using a node dependent logic 132 for processing the requests." (Col. 8, lines 49-52 - emphasis added)
>
> "FIG. 8 illustrates the P1394.2 functional blocks of a transmission interface that may be used in a computing node for a multiprocessor array used in the present invention. The computing array has an east to west macro 118 and a west to east macro 120 which are connected by a processor bus 119 to a processing unit (not shown). The terms "east" and "west" are used for convenience to indicate two opposite directions in a ring topology." (Col. 8, lines 29-36 - not specifically cited - emphasis added)

Appellant respectfully disagrees and asserts that the excerpt from Irwin relied upon by the Examiner simply teaches that "received requests are forwarded to a processing unit (not shown) via a processor bus 119" (emphasis added). Further, Irwin teaches that "[t]he computing array has an east to west macro 118 and a west to east macro 120 which are connected by a processor bus 119 to a processing unit (not shown)" (Col. 8, lines 31-34 – emphasis added). However, the mere disclosure of a processing unit connected by a processor bus to an east to west macro and a west to east macro, as in Irwin, fails to teach "a data bus connecting the addressable memory unit and the first and second data processing units" (emphasis added), as claimed by appellant.

The Examiner is reminded that a claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described in a single prior art reference. Verdegaal Bros. v. Union Oil Co. Of California, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). Moreover, the identical invention must be shown in as complete detail as contained in the claim. Richardson v. Suzuki Motor Co.868 F.2d 1226, 1236, 9USPQ2d 1913, 1920 (Fed. Cir. 1989). The elements must be arranged as required by the claim.

This criterion has simply not been met by the Irwin reference, especially in view of the arguments made hereinabove.

Issue #2:

The Examiner has rejected Claims 2-16, and 30 under 35 U.S.C. 103(a) as being unpatentable over Irwin (U.S. Patent No. 6,393,026), in view of Kadambi et al. (U.S. Patent No. 6,850,521).

Group #1: Claims 2 and 16

Appellant respectfully asserts that such claims are not met by the prior art for the reasons argued with respect to Issue #1, Group #1.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine

reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on appellant's disclosure. *In re Vaeck*,947 F.2d 488, 20 USPQ2d 1438 (Fed.Cir.1991).

Appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #2: Claims 3, 5, 13 and 14

With respect to the Claim 3, the Examiner has relied on the following excerpts to make a prior art showing of appellant's claimed "policy action table connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy."

> "FIG. 1 shows an example of a router having forwarding engines at network interfaces. The router 10 consists of a plurality of network interfaces. 12 that are interconnected through a switching fabric 14 that is controlled through a network processor 1G. Each interface 12 may be provisioned with a forwarding engine 18 that processes incoming packets in order to determine which outgoing network interface 12 needs to be accessed through the interconnect structure of the switching fabric 14. By locating the forwarding algorithms at the interface 12, the forwarding engine 18 is required to operate at the wire rate defined by the associated transmission interface. To allow for processing suitable for transmission facilities having wire speeds of OC-12, OC-48, and OC-192, a local version of the forwarding algorithms and data tables found normally in the network processor are downloaded from the network processor 16 to the forwarding engine 18 of the network interface 12. The forwarding engine 18 is optimized for forwarding speed whereas the routing table data management operates at a lower rate of changes occurring to the network configuration. Since the network processor 16 is no longer required to make routing decisions for a data packet, the packet throughput is capable of being scaled with the number of physical interfaces 12. The single point of congestion as formerly experienced with the centralization of the routing algorithms is replaced by multiple processors 12." (Irwin - Col. 1, lines 32-57 - emphasis added)

> "Referring once again to FIG. 14, after. FFP 141 applies appropriate configured filters and results are obtained from the appropriate rules

> table 22, logic 1411 in FFP 141 determines and takes the appropriate action. The filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers." (Kadambi - Col. 35, lines 57-64 - emphasis added)

Appellant respectfully asserts that the excerpt from Irwin relied upon by the Examiner merely teaches that "a local version of the forwarding algorithms and data tables found normally in the network processor are downloaded from the network processor 16 to the forwarding engine 18 of the network interface 12" and that "the routing table data management operates at a lower rate of changes occurring to the network configuration" (emphasis added). However, merely teaching that data tables are downloaded from the network processor to the forwarding engine, and that the routing table data management operates at a lower rate, as in Irwin, simply fails to suggest a "policy action table," much less a "policy action table connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy" (emphasis added), as claimed by appellant. Appellant notes that even the Examiner has admitted that "Irwin fails to teach [such] limitation."

Furthermore, appellant respectfully asserts that the excerpt form Kadambi relied upon by the Examiner simply teaches that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (emphasis added). However, the mere disclosure of filtering logic that can discard the packet, send the packet to the CPU, modify the packet header, and recalculate any IP checksum fields, as in Kadambi, fails to teach a "policy action table," let alone a "policy action table connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy" (emphasis added), as claimed by appellant. Clearly, filtering logic, as in Kadambi, simply fails to even suggest a "policy action table [that] stores at least one data processing policy" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #3: Claim 4

With respect to Claim 4, the Examiner has relied on the following excerpt from Kadambi to make a prior art showing of appellant's claimed "first address pointer element for identifying the location in said addressable memory unit of one of said plurality of instruction sets." Specifically, the Examiner has argued that "Kadambi discloses logic 1411 in the FFP 141 which points the instruction sets to take action."

> "Referring once again to FIG. 14, after. FFP 141 applies appropriate configured filters and results are obtained from the appropriate rules table 22, logic 1411 in FFP 141 determines and takes the appropriate action. The filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers." (Col. 35, lines 57-64 - emphasis added)

Appellant respectfully disagrees and asserts that the excerpt from Kadambi relied upon by the Examiner teaches that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (emphasis added). However, the mere disclosure of filtering logic that can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, fails to suggest a "first address pointer element," much less "a first address pointer element for identifying the location in said addressable memory unit of one of said plurality of instruction sets" (emphasis added), as claimed by appellant.

Further, with respect to Claim 4, the Examiner has relied on the following excerpt from Kadambi to make a prior art showing of appellant's claimed "second address pointer element for identifying the location in said addressable memory unit of a state block." Specifically, the Examiner has argued that "Kadambi discloses the FFP which is essentially a state machine."

> "FFP 141 is essentially a state machine driven programmable rules engine. The filters used by the FFP in a first embodiment are 64 (sixty-four) bytes wide, and are applied on an incoming packet. In some embodiments, a 64 byte filter mask can be used and applied to any selected 64 bytes or 512 bits of a packet. In another embodiment, however, a filter can be created by parsing selected fields of an

> incoming packet such that a 64 byte filter mask is created, which will be selectively applied to fields of interest of an incoming packet. In yet another embodiment, a filter can be created by applying a predetermined number of offsets to the incoming data packet 112, wherein a predetermined number of bytes immediately following each individual offset are parsed from the packet and thereafter concatenated together to form a filter value utilized in the filtration process." (Col. 31, lines 20-34 - emphasis added)

Appellant respectfully disagrees and asserts that the excerpt from Kadambi relied upon by the Examiner simply teaches that "FFP 141 is essentially a state machine driven programmable rules engine" (emphasis added). However, the mere disclosure of a state machine driven programmable rules engine, as in Kadambi, simply fails to suggest any sort of a "second address pointer element," much less "a second address pointer element for identifying the location in said addressable memory unit of a state block" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group II4: Claims 6-12

With respect to Claim 6, the Examiner has relied on Col. 35, lines 24-65 from Kadambi to make a prior art showing of appellant's claimed technique "wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table."

Appellant respectfully asserts that the excerpt from Kadambi relied upon by the Examiner merely teaches that "a logical AND operation is performed with the filter mask, having the selected fields enabled, and the packet" such that "[i]f there is a match, the matching entries are applied to rules tables 22, in order to determine which specific actions will be taken" (Col. 35, lines 24-28 – emphasis added). Further, the excerpt from Kadambi teaches that "since particular rules must be applied for various types of packets, the rules table requirements are minimized by setting all incoming packets to be 'tagged' packets" (Col. 35, lines 29-32 – emphasis added).

Additionally, the excerpt from Kadambi teaches that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (Col. 35, lines 59-64 – emphasis added).

However, merely teaching that if there is a match, then the matching entries are applied to the rules table in order to determine which specific actions will be taken, in addition to suggesting that filtering logic can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, simply fails to suggest a technique "wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant.

Clearly, applying matching entries to the rules table to determine which specific actions will be taken, in addition to filtering logic determining and taking the appropriate actions, as in Kadambi, simply fails to even suggest "generating a first thread," much less "generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant. Furthermore, suggesting that the rules table requirements are minimized by setting all incoming packets to be 'tagged' packets, as in Kadambi, also simply fails to suggest "generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #5: Claim 15

With respect to Claim 15, the Examiner has relied on Col. 31, lines 34-45 and Col. 35, lines 24-65 of Kadambi to make a prior art showing of appellant's claimed "memory unit connected to said first data processing unit and to said second data processing unit, said memory unit adapted to temporarily store packets before processing by said second data processing unit." Specifically, the Examiner has argued that "Kadambi discloses packets stored within FFP."

Appellant respectfully disagrees and asserts that the excerpts from Kadambi relied upon by the Examiner teach that "FFP 141 includes filtering logic 1411, ...which selectively parses predetermined fields from the incoming data packets, thereby effectively obtaining the values of the desired fields from the MAC, IP, TCP, and UDP headers" (Col. 31, lines 41-45 – emphasis added). Further, the excerpts teach that "FFP 141 is shown to include filter database 1410 containing filter masks therein" (Col. 35, lines 46-47 – emphasis added).

Therefore, in Kadambi, the FFP is taught to include filtering logic that selectively parses predetermined fields from incoming data packets, and a filter database containing filter masks, which simply fails to meet appellant's claimed "memory unit connected to said first data processing unit and to said second data processing unit, said memory unit adapted to temporarily store packets before processing by said second data processing unit" (emphasis added), as claimed by appellant. Clearly, the FFP including filtering logic and a filter database, as in Kadambi, simply fails to meet "said memory unit adapted to temporarily store packets before processing by said second data processing unit" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Group #6: Claim 30

With respect to Claim 30, the Examiner has relied on the following excerpts from Irwin to meet appellant's claimed "first data processing unit adapted to filter incoming packets."

"The data packet processing is performed by executing a packet switching software of the router. The processing for forwarding a data packet can be broken into a number of procedures. Typically, packet forwarding requires procedures, such as translation of a header of the data packet to identify suitable routes, classification of a packet flow according to source and destination addresses..." (Col. 5, lines 46-52 - emphasis added)

"[serv]ice route pruning, metric route pruning, policy route pruning, option processing, congestion control, scheduling, and performance monitoring." (Col. 5, lines 53-55 - not specifically cited - emphasis added)

"As each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system." (Col. 6, lines 5-8)

Appellant respectfully asserts that the excerpts from Irwin relied upon by the Examiner merely disclose that "[t]he processing for forwarding a data packet can be broken into a number of procedures such as translation of a header of the data packet to identify suitable routes, classification of a packet flow according to source and destination addresses, type of service route pruning, metric route pruning, policy route pruning, option processing, congestion control, scheduling, and performance monitoring" (emphasis added). However, merely disclosing that forwarding a data packet may be broken into procedures such as the translation of a header to identify suitable routes, and classification of a packet flow according to source and destination addresses, as in Irwin, fails to suggest "a first data processing unit adapted to filter incoming packets" (emphasis added), as claimed by appellant. Clearly, procedures for forwarding a data packet, as in Irwin, simply fails to even suggest "filter[ing] incoming packets," as claimed by appellant.

Additionally, appellant respectfully asserts that it would be unobvious to modify Irwin to "filter incoming packets" (emphasis added), as claimed by appellant. Specifically, Irwin teaches "a router having forwarding engines at network interfaces" where "[e]ach interface 12 may be provisioned with a forwarding engine 18 that processes incoming packets in order to determine which outgoing network interface 12 needs to be accessed through the interconnect structure of the switching fabric 14" (Col. 1, lines 32-40 – emphasis added). Further, Irwin teaches that "[s]ince the network processor 16 is no longer required to make routing decisions for a data packet, the packet throughput is capable of being scaled with the number of physical interfaces 12" such that "[t]he single point of congestion as formerly experienced with the centralization of

the routing algorithms is replaced by multiple processors 12" (Col. 1, lines 52-57 – emphasis added).

Appellant asserts that Irwin's router with forwarding engines at each network interface simply utilizes the forwarding engines to determine which outgoing network interface needs to be accessed through the interconnect structure in order to alleviate the network processor from making routing decisions, which allows the throughput to be scaled with each interface. Clearly, a router with a forwarding engine at each interface that determines which outgoing network interface needs to be accessed through the interconnect structure, would be unable to filter incoming packets due to the congestion as formerly experienced with the centralization of the routing algorithms, as expressly noted in Irwin. Therefore, appellant asserts that it would be unobvious to include filtering, in the manner claimed by appellant, in Irwin's multiple network interface router.

Further, with respect to Claim 30, the Examiner has relied on the following excerpt from Irwin to meet appellant's claimed "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit."

> "As each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system. The program counter uniquely defines a main program flow in the main forwarding program that is executed for the packet to select an output queue of the router for routing the packet. Stepping through the program counter temporarily creates a virtual CPU execution unit, or a hardware thread, for the data packet. Each thread is represented by a thread identification number which is defined by the program counter and a register file used by the process followed for packet processing." (Col. 6, lines 5-16 - emphasis added)

Appellant respectfully asserts that the excerpt from Irwin relied upon by the Examiner merely teaches that "[a]s each data packet enters the data packet processing system, a program counter is assigned to the packet for the duration of the time the packet is held in the data packet processing system" and that "[s]tepping through the program counter temporarily creates a virtual CPU execution unit, or a hardware thread, for the data packet" (emphasis added).

However, the mere disclosure of assigning a program counter to the packet, such that stepping through the program counter creates a virtual CPU execution unit or hardware thread for the data packet, as in Irwin, fails to teach "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit" (emphasis added), as claimed by appellant. Clearly, stepping through the program counter to create a hardware thread for the packet, as in Irwin, simply fails to even suggest "process[ing] incoming packets according to one of said plurality of instruction sets **after the filtering**, based on a thread" (emphasis added), as claimed by appellant.

Additionally, appellant respectfully asserts that Irwin, as argued above, simply fails to suggest any sort of "filtering," as claimed by appellant, let alone in the specific context claimed by appellant. For example, appellant asserts that Irwin merely discloses that "[w]hen the end procedural call is read and the end of the main forwarding program has been reached (S32), if there is no data packet to be processed (S34), the thread is placed in the queue of idle threads since the packet processing has been completed and the packet has been forwarded to the designated route" (Col. 10, lines 56-61 – emphasis added). Further, Irwin teaches that "[i]f there is a data packet to be processed (S34), the thread is reactivated for a new packet (S02)" (Col. 10, lines 61-62).

However, disclosing that the thread is placed in the queue of idle threads after the packet processing has been completed and the packet has been forwarded to the designated route, as in Irwin, fails to suggest "a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit" (emphasis added), as claimed. Clearly, placing the thread in the queue of idle threads after processing is completed, as in Irwin, simply fails to even suggest "process[ing] incoming packets according to one of said plurality of instruction sets **after the filtering**, **based on a thread assigned** to the incoming packets by the first data processing unit" (emphasis added), as claimed by appellant.

In addition, with respect to Claim 30, the Examiner has relied on the following excerpts from Irwin to meet appellant's claimed "a data bus connecting the addressable memory unit and the first and second data processing units."

> "The received requests are forwarded to a processing unit (not shown) via a processor bus 119 using a node dependent logic 132 for processing the requests." (Col. 8, lines 49-52 - emphasis added)
>
> "FIG. 8 illustrates the P1394.2 functional blocks of a transmission interface that may be used in a computing node for a multiprocessor array used in the present invention. The computing array has an east to west macro 118 and a west to east macro 120 which are connected by a processor bus 119 to a processing unit (not shown). The terms "east" and "west" are used for convenience to indicate two opposite directions in a ring topology." (Col. 8, lines 29-36 - not specifically cited - emphasis added)

Appellant respectfully asserts that the excerpt from Irwin relied upon by the Examiner simply teaches that "received requests are forwarded to a processing unit (not shown) via a processor bus 119" (emphasis added). Further, Irwin teaches that "[t]he computing array has an east to west macro 118 and a west to east macro 120 which are connected by a processor bus 119 to a processing unit (not shown)" (Col. 8, lines 31-34 – emphasis added). However, the mere disclosure of a processing unit connected by a processor bus to an east to west macro, and a west to east macro, as in Irwin, fails to teach "a data bus connecting the addressable memory unit and the first and second data processing units" (emphasis added), as claimed by appellant.

Additionally, with respect to Claim 30, the Examiner has relied on the following excerpt to make a prior art showing of appellant's claimed technique "wherein a policy action table is connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy."

> "Referring once again to FIG. 14, after. FFP 141 applies appropriate configured filters and results are obtained from the appropriate rules table 22, logic 1411 in FFP 141 determines and takes the appropriate action. The filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers. The modification occurs at buffer slicer 144, and the packet is placed on C channel 81." (Kadambi - Col. 35, lines 57-64 - emphasis added)

Appellant respectfully asserts that the excerpt form Kadambi relied upon by the Examiner simply teaches that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (emphasis added). However, the mere disclosure of filtering logic that can discard the packet, send the packet to the CPU, modify the packet header, and recalculate any IP checksum fields, as in Kadambi, fails to teach a "policy action table," much less that "a policy action table is connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy" (emphasis added), as claimed by appellant. Clearly, filtering logic, as in Kadambi, simply fails to even suggest a "policy action table [that] stores at least one data processing policy" (emphasis added), as claimed by appellant.

Furthermore, with respect to Claim 30, the Examiner has relied on Col. 35, lines 24-65 from Kadambi to make a prior art showing of appellant's claimed technique "wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table."

Appellant respectfully asserts that the excerpt from Kadambi relied upon by the Examiner merely teaches that "a logical AND operation is performed with the filter mask, having the selected fields enabled, and the packet" such that "[i]f there is a match, the matching entries are applied to rules tables 22, in order to determine which specific actions will be taken" (Col. 35, lines 24-28 – emphasis added). Further, the excerpt from Kadambi teaches that "since particular rules must be applied for various types of packets, the rules table requirements are minimized by setting all incoming packets to be 'tagged' packets" (Col. 35, lines 29-32 – emphasis added). Additionally, the excerpt from Kadambi teaches that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (Col. 35, lines 59-64 – emphasis added).

However, merely teaching that if there is a match, then the matching entries are applied to the rules table in order to determine which specific actions will be taken, in addition to suggesting that filtering logic can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, simply fails to suggest a technique "wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant.

Clearly, applying matching entries to the rules table to determine which specific actions will be taken, in addition to filtering logic determining and taking the appropriate actions, as in Kadambi, simply fails to even suggest "generating a first thread," much less "generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant. Furthermore, suggesting that the rules table requirements are minimized by setting all incoming packets to be 'tagged' packets, as in Kadambi, also simply fails to suggest "generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table" (emphasis added), as claimed by appellant.

Still yet, with respect to Claim 30, the Examiner has relied on Col. 31, lines 20-34 and Col. 35, lines 57-64 from Kadambi to make a prior art showing of appellant's claimed technique "wherein said data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in said addressable memory unit, said state block used by said first set of instructions for processing the first incoming packet."

Appellant respectfully asserts that the Col. 35, lines 57-64 from Kadambi relied upon by the Examiner teaches that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate

action with respect to the headers" (emphasis added). However, the mere disclosure of filtering logic that can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, fails to suggest a "first address pointer...and a second address pointer," much less a technique "wherein said data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in said addressable memory unit, said state block used by said first set of instructions for processing the first incoming packet" (emphasis added), as claimed by appellant.

In addition, appellant respectfully asserts that Col. 31, lines 20-34 from Kadambi relied upon by the Examiner simply teaches that "FFP 141 is essentially a state machine driven programmable rules engine" (emphasis added). However, the mere disclosure of a state machine driven programmable rules engine, as in Kadambi, simply fails to suggest any sort of a "first address pointer...and a second address pointer," let alone a technique "wherein said data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in said addressable memory unit, said state block used by said first set of instructions for processing the first incoming packet" (emphasis added), as claimed by appellant.

Additionally, with respect to Claim 30, the Examiner has relied on Col. 35, lines 24-65 from Kadambi to make a prior art showing of appellant's claimed technique "wherein said first processing unit further comprises logic for matching a second incoming packet to a stored second rule and for generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table."

Appellant respectfully asserts that the excerpt from Kadambi relied upon by the Examiner merely teaches that "a logical AND operation is performed with the filter mask, having the selected fields enabled, and the packet" such that "[i]f there is a match, the matching entries are applied to rules tables 22, in order to determine which specific actions will be taken" (Col. 35, lines 24-28 – emphasis added). Further, the excerpt from Kadambi teaches that "since particular rules must be applied for various types of packets, the rules table requirements are minimized by

setting all incoming packets to be 'tagged' packets" (Col. 35, lines 29-32 – emphasis added). Additionally, the excerpt from Kadambi teaches that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (Col. 35, lines 59-64 – emphasis added).

However, merely teaching that if there is a match, then the matching entries are applied to the rules table in order to determine which specific actions will be taken, in addition to suggesting that filtering logic can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, simply fails to suggest a technique "wherein said first processing unit further comprises logic for matching a second incoming packet to a stored second rule and for generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table" (emphasis added), as claimed by appellant.

Clearly, applying matching entries to the rules table to determine which specific actions will be taken, in addition to filtering logic determining and taking the appropriate actions, as in Kadambi, simply fails to even suggest "generating a second thread," much less "generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table" (emphasis added), as claimed by appellant. Furthermore, suggesting that the rules table requirements are minimized by setting all incoming packets to be 'tagged' packets, as in Kadambi, also simply fails to suggest "generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table" (emphasis added), as claimed by appellant.

Moreover, with respect to Claim 30, the Examiner has relied on the following excerpt from Kadambi to make a prior art showing of appellant's claimed technique "wherein a memory unit is connected to said first data processing unit and to said second data processing unit, said

memory unit adapted to temporarily store packets before processing by said second data processing unit."

> "FIG. 2 illustrates a more detailed block diagram of the functional elements of SOC 10. As evident from FIG. 2 and as noted above, SOC 10 includes a plurality of modular systems on-chip, with each modular system, although being on the same chip, being functionally separate from the other modular systems. Therefore, each module can efficiently operate in parallel with other modules, and this configuration enables a significant amount of freedom in updating and re-engineering SOC 10." (Col. 5, lines 46-54 - emphasis added)

Appellant respectfully asserts that the excerpt from Kadambi relied upon by the Examiner teaches that "SOC 10 includes a plurality of modular systems on-chip" and that "each module can efficiently operate in parallel with other modules" (emphasis added). However, the mere disclosure of a SOC including a plurality of modular systems that can operate in parallel with other modules, as in Kadambi, simply fails to meet appellant's claimed technique "wherein a memory unit is connected to said first data processing unit and to said second data processing unit, said memory unit adapted to temporarily store packets before processing by said second data processing unit" (emphasis added), as claimed by appellant.

In addition, with respect to Claim 30, the Examiner has relied upon Col. 31, lines 35-45 and Col. 35, lines 24-65 from Kadambi to make a prior art showing of appellant's claimed technique "wherein the apparatus includes a control logic unit couples to an input and the policy condition table for feeding an arithmetic logic unit, which is in turn coupled to the policy action table and the state block for generating an output."

Appellant respectfully asserts that the excerpts from Kadambi relied upon by the Examiner teach that "FFP 141 includes filtering logic 1411, ...which selectively parses predetermined fields from the incoming data packets, thereby effectively obtaining the values of the desired fields from the MAC, IP, TCP, and UDP headers" (Col. 31, lines 41-45 – emphasis added). Further, the excerpts teach that "FFP 141 is shown to include filter database 1410 containing filter masks therein" (Col. 35, lines 46-47 – emphasis added). However, a FFP including filtering logic that selectively parses predetermined fields from incoming data packets, and a filter database containing filter masks, as in Kadambi, simply fails to suggest a technique "wherein the apparatus includes a control logic unit couples to an input and the policy condition table for

feeding an arithmetic logic unit, which is in turn coupled to the policy action table and the state block for generating an output" (emphasis added), as claimed by appellant.

Additionally, the excerpts from Kadambi teach that "logic 1411 in FFP 141 determines and takes the appropriate action," and that "[t]he filtering logic can discard the packet, send the packet to the CPU 52, modify the packet header or IP header, and recalculate any IP checksum fields or takes other appropriate action with respect to the headers" (Col. 35, lines 59-64 – emphasis added). However, the mere disclosure of filtering logic that can determine and take the appropriate action such as discarding the packet, sending the packet to the CPU, modifying the packet header, and recalculating any IP checksum fields, as in Kadambi, simply fails to suggest a technique "wherein the apparatus includes a control logic unit couples to an input and the policy condition table for feeding an arithmetic logic unit, which is in turn coupled to the policy action table and the state block for generating an output" (emphasis added), as claimed by appellant.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

Issue #3:

The Examiner has rejected Claims 17, and 18 under 35 U.S.C. 103(a) as being unpatentable over Kadambi et al. (U.S. Patent No. 6,850,521), in view of Scales (U.S. Patent No. 5,761,729).

Group #1: Claims 17 and 18

Appellant respectfully asserts that such claims are not met by the prior art for the reasons argued with respect to Issue #2, Group #1.

Again, appellant respectfully asserts that at least the third element of the *prima facie* case of obviousness has not been met, since the prior art excerpts, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above.

In view of the remarks set forth hereinabove, all of the independent claims are deemed allowable, along with any claims depending therefrom.

VIII CLAIMS APPENDIX (37 C.F.R. § 41.37(c)(1)(viii))

The text of the claims involved in the appeal (along with associated status information) is set forth below:

1. (Previously Presented) An apparatus for processing data packets, comprising:

a first data processing unit adapted to filter incoming packets;

an addressable memory unit in which a plurality of instruction sets for packet processing are stored;

a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit; and

a data bus connecting the addressable memory unit and the first and second data processing units.

2. (Original) The apparatus of claim 1, further comprising a policy condition table connected to said first data processing unit, said policy condition table having a plurality of rules stored therein.

3. (Original) The apparatus of claim 1, further comprising a policy action table connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy.

4. (Original) The apparatus of claim 3, wherein at least one of said policies comprises:

a first address pointer element for identifying the location in said addressable memory unit of one of said plurality of instruction sets, and

a second address pointer element for identifying the location in said addressable memory unit of a state block.

5. (Original) The apparatus of claim 3, wherein said first data processing unit assigns a thread to each said incoming packet, wherein said thread corresponds to one of said policies stored in said policy action table.

6. (Original) The apparatus of claim 3, wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table.

7. (Original) The apparatus of claim 6, wherein said second data processing unit is adapted to process the first incoming packet according to said data processing policy corresponding to said first thread.

8. (Original) The apparatus of claim 6, wherein said data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in said addressable memory unit, said state block used by said first set of instructions for processing the first incoming packet.

9. (Original) The apparatus of claim 6, wherein said thread is assigned to said first incoming packet based on said first rule.

10. (Original) The apparatus of claim 6, wherein said first processing unit further comprises logic for matching a second incoming packet to a stored second rule and for generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table.

11. (Original) The apparatus of claim 10, wherein said second data processing unit is adapted to process the second incoming packet according to said data processing policy corresponding to said second thread.

12. (Original) The apparatus of claim 10, wherein said second thread is assigned to said second incoming packet based on said second rule.

13. (Previously Presented) The apparatus of claim 3, wherein said first processing unit further comprises logic for matching a plurality of incoming packets to a stored corresponding

plurality of rules and for generating a thread for each packet that matches one of said plurality of rules, each said thread identifying the location of one of said at least one data processing policy in said policy action table.

14. (Original) The apparatus of claim 13, wherein the second data processing unit is adapted to process each packet according to said data processing policy corresponding to said thread associated with said packet.

15. (Original) The apparatus of claim 13, further comprising a memory unit connected to said first data processing unit and to said second data processing unit, said memory unit adapted to temporarily store packets before processing by said second data processing unit.

16. (Original) The apparatus of claim 1, wherein said second data processing unit comprises a plurality of general purpose processors for executing instructions in parallel.

17. (Original) The apparatus of claim 16, wherein at least one said general purpose processor comprises a complex instruction set computer processor.

18. (Original) The apparatus of claim 16, wherein at least one said general purpose processor comprises a reduced instruction set computer processor.

19. - 29. (Cancelled)

30. (Previously Presented) An apparatus for processing data packets, comprising:

a first data processing unit adapted to filter incoming packets;

an addressable memory unit in which a plurality of instruction sets for packet processing are stored;

a second data processing unit adapted to process incoming packets according to one of said plurality of instruction sets after the filtering, based on a thread assigned to the incoming packets by the first data processing unit; and

a data bus connecting the addressable memory unit and the first and second data processing units;

wherein a policy condition table is connected to said first data processing unit, said policy condition table having a plurality of rules stored therein;

wherein a policy action table is connected to said data bus and said addressable memory unit, wherein said policy action table stores at least one data processing policy;

wherein said first data processing unit comprises logic for matching a first incoming packet to a stored first rule and for generating a first thread if the first incoming packet matches said first rule, said first thread identifying the location of one of said at least one data processing policies in said policy action table;

wherein said second data processing unit is adapted to process the first incoming packet according to said data processing policy corresponding to said first thread;

wherein said data processing policy comprises a first address pointer to a starting address of a first set of instructions and a second address pointer to a starting address of a state block stored in said addressable memory unit, said state block used by said first set of instructions for processing the first incoming packet;

wherein said first processing unit further comprises logic for matching a second incoming packet to a stored second rule and for generating a second thread if the second incoming packet matches the second rule, said second thread identifying the location of one of said at least one data processing policy in said policy action table;

wherein said second data processing unit is adapted to process the second incoming packet according to said data processing policy corresponding to said second thread;

wherein a memory unit is connected to said first data processing unit and to said second data processing unit, said memory unit adapted to temporarily store packets before processing by said second data processing unit;

wherein said second data processing unit comprises a plurality of general purpose processors for executing instructions in parallel;

wherein the apparatus includes a control logic unit coupled to an input and the policy condition table for feeding an arithmetic logic unit, which is in turn coupled to the policy action table and the state block for generating an output.

IX EVIDENCE APPENDIX (37 C.F.R. § 41.37(c)(1)(ix))

There is no such evidence.

X RELATED PROCEEDING APPENDIX (37 C.F.R. § 41.37(c)(1)(x))

N/A

In the event a telephone conversation would expedite the prosecution of this application, the Examiner may reach the undersigned at (408) 971-2573. For payment of any additional fees due in connection with the filing of this paper, the Commissioner is authorized to charge such fees to Deposit Account No. 50-1351 (Order No. NAI1P069).

Respectfully submitted,

By: /KEVINZILKA/ Date: January 22, 2008
Kevin J. Zilka
Reg. No. 41,429

Zilka-Kotab, P.C.
P.O. Box 721120
San Jose, California 95172-1120
Telephone: (408) 971-2573
Facsimile: (408) 971-4660